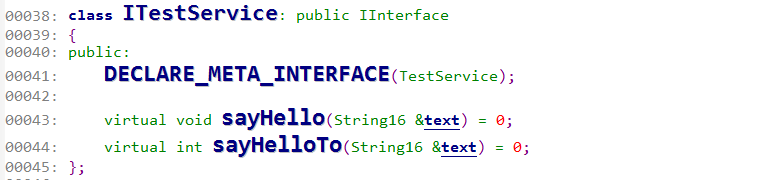
一、binder服务实现步骤：

1、定义一个Service类继承IInterface:



（1）Service类名命令规则：IXXXX，类名必须以I开始，后面的XXXX表示Service的名字；

（2）申明元数据接口：

DECLARE\_META\_INTERFACE(XXXX);

XXXX是类名出掉前面的I。

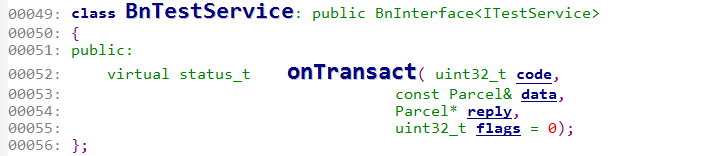
（3）类中其它成员函数：表示此服务提供的接口，必须声明虚函数，后续需要在server和client类中同时实现对应的方法。

2、实现元数据接口：

IMPLEMENT\_META\_INTERFACE(XXXX, "YYYY");

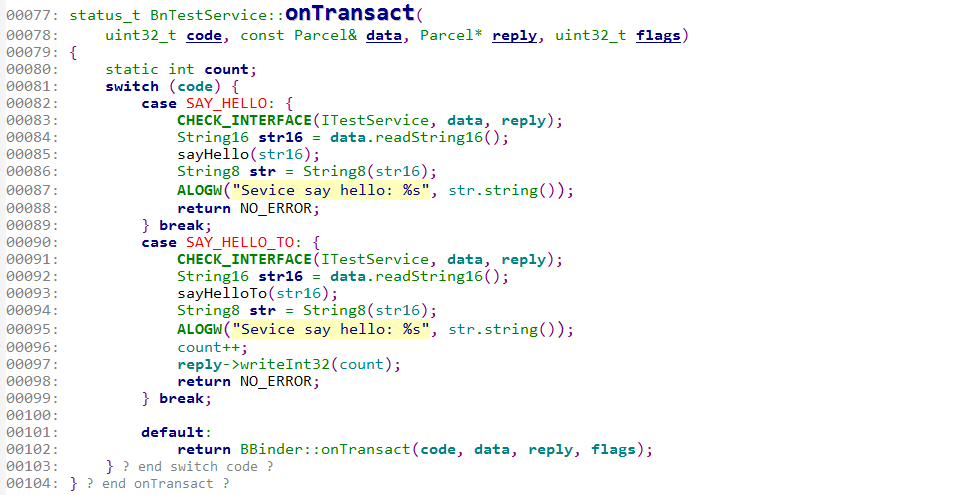
XXXX是类名出掉前面的I，YYYY表示服务描述字段。

3、定义一个BnXXXX类继承BnInterface<IXXXX>

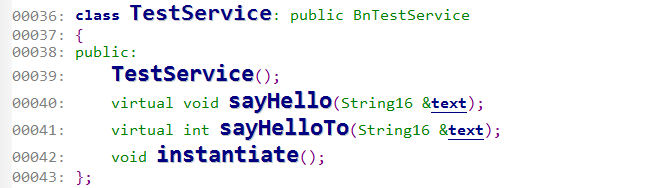


BnXXXX是一个可以直接使用的服务类，也可以通过定义子类的方式使用。如果需要直接使用BnXXXX类，需要实现IXXXX中的虚函数。

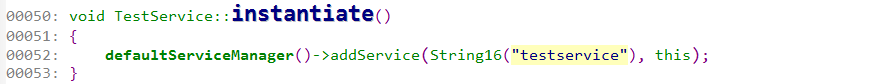
其中onTransact函数是服务入口函数，所有来自client程序的远程调用，服务程序都会回调onTransact函数，通过不同code调用不同的成员函数（）；



4、使用BnXXXX子类定义服务



需要定义一个服务注册函数：instantiate（函数名可以自定义）



实例化父类中的所用虚函数。

5、创建服务进程



（1）添加服务：testService.instantiate();

（2）阻塞服务进程，读取client的请求：

下面两个函数会进入阻塞。

proc->startThreadPool();

IPCThreadState::self()->joinThreadPool(false);

startThreadPool会创建一个新线程，并阻塞读取client的请求。

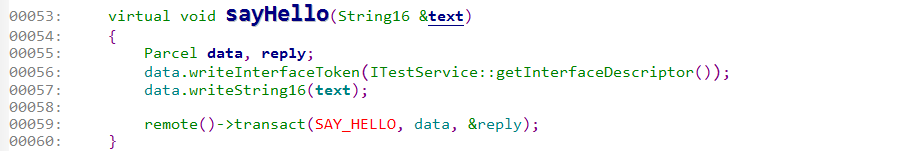
joinThreadPool是将当前主线程阻塞，同样可以处理client的请求，这样可以防止进程退出。

二、binder服务使用步骤：

1、定义BpXXXX类继承BpInterface<IXXXX>类



2、使用IXXXX中的虚函数：



构造远程远程调用的参数和返回值，调用remote()->transact(SAY\_HELLO, data, &reply);实现远程调用。

3、使用BpXXXX类发起远程调用：



（1）获取ServiceManagerde 的实例：

sp<IServiceManager> sm = defaultServiceManager();

（2）获取服务：

binder = sm->getService(String16("testservice"));

（3）实例化BpXXXX类：

testService = interface\_cast<ITestService>(binder);

（4）发起远程调用：

testService->sayHello(hello1);

二、关键类继承关系

BnTestService --> BnInterface--> ITestService,BBinder

BpTestService-->BpInterface-->ITestService, BpRefBase

ITestService--> IInterface

BBinder--> IBinder--> RefBase

BpRefBase--> RefBase